Healthcare Analytics with SQL

BADM – Project 3

Task 1 – Inner and Equi Joins

I wrote a query to retrieves the names of patients and doctors, their specialization, and appointment status, but only for appointments where the status is 'Completed’.

It will joins the appointments table with patients and doctors table using their IDs.

Result:

There are 1000 appointments which are in completed status

Task 2 – Left Join with Null handling

Here I wrote a query to list a patients who have never had an appointment, showing their name, contact number, and address. It uses a LEFT JOIN and filters where no appointment exists.

It will joins the appointments table with patients table using patient ID.

Result:

There are 662 patients does not made any appointments yet

Task 3 – Right Join and Aggregate Functions

This query gets each doctor's name, specialization, and the number of diagnoses they made.

- Used a RIGHT JOIN to include all doctors, even those with zero diagnoses.
- Counted how many times each doctor appears in the diagnoses table.
- Groups the result by doctor to get totals.

Result:

300 doctors are available with different specializations. It will also gives the number of diagnoses made

Task 4 – Full Join for Overlapping Data

Since MySQL does not support FULL JOIN directly, I have used multiple LEFT JOINs to simulate FULL JOIN.

This query retrieves **all diagnoses** along with related **appointments**, **patients**, and **doctors.**

Starts from the diagnoses table and LEFT JOINs with tables Appointments, Patients and Doctors

Result:

Even if a diagnosis has no appointment, the row is still shown with NULL values in appointment fields.

Task 5 – Window Functions (Ranking and Aggregation)

This query shows the total number of appointments each patient had with each doctor, and **ranks the patients per doctor** based on their appointment count using:

- RANK() – gives gaps in ranking for ties
- DENSE_RANK() – gives continuous ranking even for ties

Result:

We ranked the patients for each doctor based on how many appointments they had, showing both total count and relative position.

Task 6 – Conditional Expressions

This query groups patients by **age ranges** (18–30, 31–50, 51+) and counts how many fall into each group.

Categorizes patients into age groups and shows the total number in each group for demographic analysis.

Result:

Out of 5000 patients, **2864** are falls into 51+ age group

1416 are from between 31-50 and **900** are from between 18-30

Task 7 – Numeric and String Functions

This query fetches patients whose **contact number ends with ‘1234’**, and converts their names to **uppercase**.

Identifies patients with contact numbers ending in ‘1234’ and formats their names in uppercase for standardized display.

Result:

We have one name found in the table with contact number ending with 1234
“PATIENT_1234, 98765431234”

Task 8 – Subqueries for Filtering

This query finds patients who have been prescribed only **‘Insulin’** and **no other medications**.

Listing patients who were treated exclusively with ‘Insulin’, helped to identify the cases with targeted medication use.

Result:

We have found 252 patients how have been prescriber only ‘Insulin’

Task 9 – Date and Time Functions

This query calculates the **average duration of medication** prescribed for each **diagnosis**.

- Calculated the number of days each medication was prescribed
- Calculated the average duration per diagnosis id
- Finally grouped the result diagnosis wise

Result:

As we calculates average prescription duration per diagnosis we come to know the treatment length and consistency.

Task 10 – Complex Joins and Aggregation

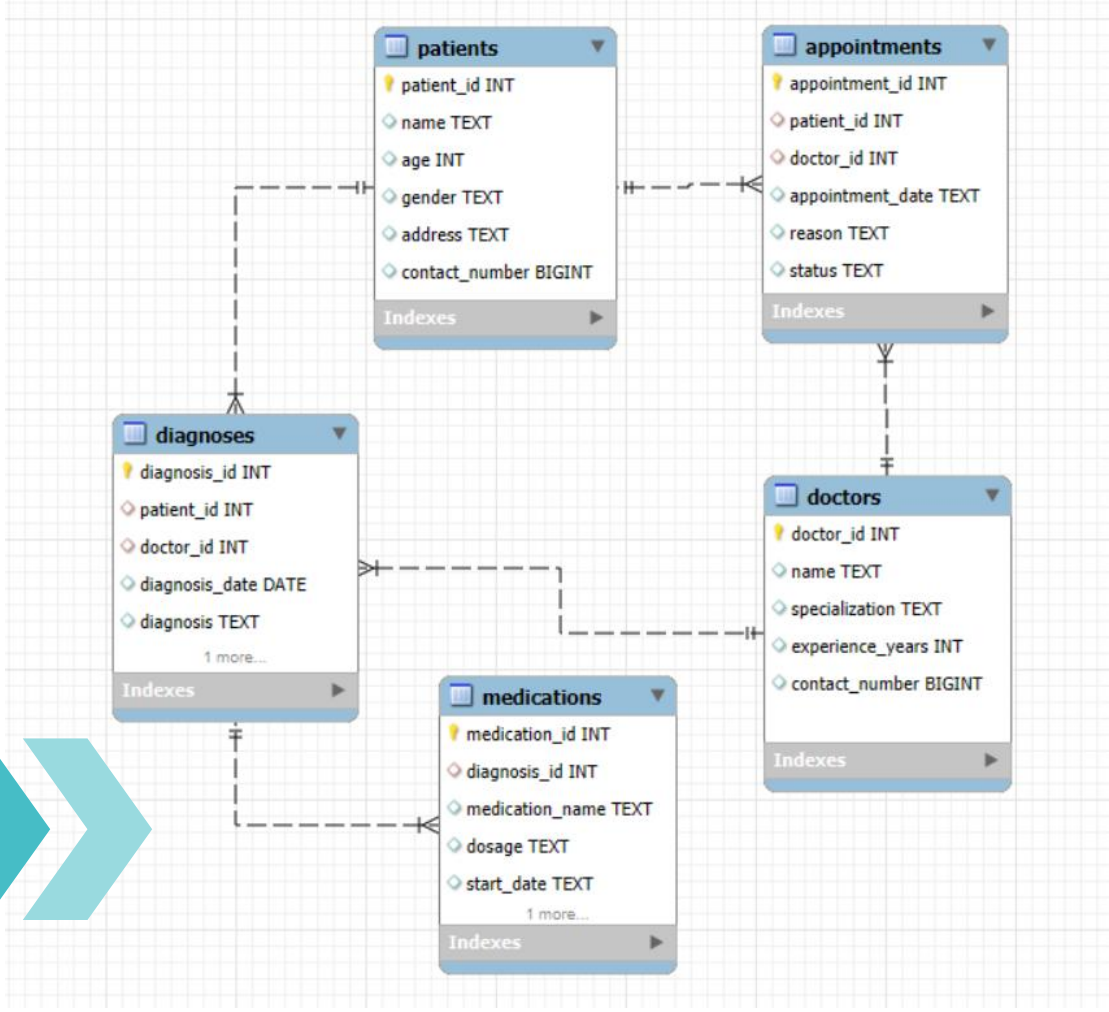
This query will find the **doctor** who has treated the highest number of **unique patients** based on the appointment records.

- Joined appointments to it’s corresponding doctor
- Count unique patients per doctor and aggregate the count per doctor
- Sorted the doctors by number of unique patients in descending order and limited 1 to get only the top 1 doctor

Result:

Doctor_37 is the one who treated the highest number of unique patients

Entity-Relationship Diagram



- Created this ER Diagram using Reverse Engineering under Database ribbon
- This diagram expresses the logical structure of the database graphically

EER Diagram.mwb

Thank you